Announcements

Program 7 – due Thursday, Nov. 16th
Reminder:
Midterm 2 on Nov. 8th

Practice from Last Time

Get the file Nov1.py from my Box.com code directory. It has the main function written for you and stubs for 2 other functions that you will need to write.

- findAverage(numbers) – will return the average of all the numbers in the list
- countNumbers(numbers, average) - will return 2 values; it counts the number of above average and below average numbers in a list

Other String Methods

• Programs commonly need to search for substrings
• Several methods to accomplish this:
  - endswith(substring): checks if the string ends with substring
    • Returns True or False
  - startswith(substring): checks if the string starts with substring
    • Returns True or False
More String Methods

• Several methods to accomplish this (cont’d):
  – find(substring): searches for substring within the string
    • Returns lowest index of the substring, or if the substring is not contained in the string, returns -1
  – replace(substring, new_string):
    • Returns a copy of the string where every occurrence of substring is replaced with new_string

Using the find method

```python
def main():
    filename = "First Last_assignsubmission_file_lastname_firstname_prg6.py"
    print(renameFile(filename))

def renameFile(fileName):
    ind = fileName.find("file_")
    fileName = fileName[ind+5]
    return fileName

main()
```

Output:
lastname_firstname_prg6.py

String Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>endswith(substring)</td>
<td>The substring argument is a string. The method returns true if the string ends with substring.</td>
</tr>
<tr>
<td>find(substring)</td>
<td>The substring argument is a string. The method returns the lowest index in the string where substring is found. If substring is not found, the method returns -1.</td>
</tr>
<tr>
<td>replace(old, new)</td>
<td>The old and new arguments are both strings. The method returns a copy of the string with all instances of old replaced by new.</td>
</tr>
<tr>
<td>startswith(substring)</td>
<td>The substring argument is a string. The method returns true if the string starts with substring.</td>
</tr>
</tbody>
</table>

Testing, Searching, and Manipulating Strings

• You can use the in operator to determine whether one string is contained in another string
  – General format: string1 in string2
    • string1 and string2 can be string literals or variables referencing strings
• Similarly you can use the not in operator to determine whether one string is not contained in another string
Class Practice

• Write a function called `count_unique` that counts the number of unique characters in a string.
  – `count_unique("abracadabra")` returns 5.
• Write a function called `count_dups` that counts the number of back-to-back duplicated characters in a string.
  – `count_dups("balloon")` returns 2.

Finding Items in Lists with the `in` Operator

• You can use the `in` operator to determine whether an item is contained in a list.
  – General format: `item in list`
  – Returns `True` if the item is in the list, or `False` if it is not in the list.
• Similarly you can use the `not in` operator to determine whether an item is not in a list.

Example Using `in` Operator

```python
# This program demonstrates the in operator
# used with a list.
def main():
    # Create a list of product numbers.
    prod_nums = ['9415', '8587', '8143', '6688']
    # Get a product number to search for.
    search = input("Enter a product number: ")
    # Determine whether the product number is in the list.
    if search in prod_nums:
        print(search, "was found in the list.")
    else:
        print(search, "was not found in the list.")

# Call the main function.
main()
```

List Methods and Useful Built-in Functions

• `append(item)` used to add items to a list – `item` is appended to the end of the existing list.
• `index(item)` used to determine where an item is located in a list.
  – Returns the index of the first element in the list containing `item`.
  – Raises `ValueError` exception if `item` not in the list.
find() doesn’t exist for lists

- `list_var.index(item)`
- Searches left to right, returns position where found, but crashes if not found.
- Let’s build an algorithm that replicates `find()`, but works for lists (returns -1 if not found).

Example Using Append

```python
def main():
    infile = open("randomNums.txt", 'r')
    numbers = []
    for line in infile:
        numbers.append(int(line))
    print(numbers)

main()
```

Output

```
[62, 37, 54, 46, 41, 80, 27, 88, 33, 11, 61, 64, 45, 56, 9, 33, 52, 63, 24, 26, 130, 95, 62, 10, 87, 58, 69, 54, 75, 41, 22, 93, 82, 16, 92, 46, 6, 71, 85, 59, 56, 22, 3, 50, 3, 20, 54, 18, 27, 78, 17, 7, 41, 83, 28, 5, 64, 60, 92, 15, 26, 57, 39, 80, 41, 67, 56, 24, 77, 28, 19, 10, 24, 75, 53, 58, 47, 50, 18, 40, 65, 24, 58, 4, 58, 82, 40, 6, 77, 85, 86, 88, 63]
```

List Methods and Useful Built-in Functions (cont’d.)

- `insert(index, item)` used to insert `item` at position `index` in the list
- `sort()` used to sort the elements of the list in ascending order
- `remove(item)` removes the first occurrence of `item` in the list
- `reverse()` reverses the order of the elements in the list
List Methods and Useful Built-in Functions (cont’d.)

- **del statement**: removes an element from a specific index in a list
  - General format: `del list[i]`
- **min and max functions**: built-in functions that returns the item that has the lowest or highest value in a sequence
  - The sequence is passed as an argument
- **sum function**: built-in functions that returns the total of all the values in a sequence
  - The sequence is passed as an argument

Example Using `del`, `min`, `max`, and `sum` functions

```python
my_list = [5, 4, 3, 2, 50, 40, 30]
del my_list[1]
print("After Deletion:", my_list)

print("The lowest value is", min(my_list))
print("The highest value is", max(my_list))
print("The sum of values in my list is", sum(my_list))

alpha_list = ['a', 'b', 'c', 'd']
print("The lowest value is", min(alpha_list))
print("The highest value is", max(alpha_list))
```

Practice

Write a program that randomly generates 20 integers between 1 and 50, and stores them in a list. Print out the lowest and the highest numbers in your list, as well as the sum of all the numbers in the list.

Write a function that prints out sums of adjacent pairs of numbers in the list

**Hint**: You don’t need the sliding window technique; instead, use math with list indices.

Write a function that takes a list and shifts all the elements in the list one spot to the left, without using slices! (the left-most element disappears)

Example: `[1, 2, 3, 4, 5]` turns into `[2, 3, 4, 5, 1]`